Numerical Solution of the Schrödinger Equation Using a Neural Network Approach

Ante Lojić Kapetanović*, Dragan Poljak†
Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture
University of Split
Split, Croatia
*alojic00@fesb.hr, †dpoljak@fesb.hr

***Abstract*—This paper overviews the solution of the Schrödinger equation for the case of one-dimensional infinite potential well with a neural network approach. Using a single hidden layer neural network, which is proved to be a universal function approximator, and by exploiting the automatic differentiation capabilities, it is possible to achieve accurate values of the wave function and eigenvalues of the ground state. The loss function with integrated physical knowledge is set up as an unconstrained nonlinear problem and parameters of a neural network are being learnt in a completely unsupervised manner. Such a technique could potentially serve as a door opener for solving high-dimensional quantum mechanics problems, otherwise tedious to set up for standard mesh-based numerical methods.**

***Index Terms*—neural network, unsupervised learning, automatic differentiation, eigenvalue problem, Schrödinger equation**

I. Introduction

The key idea of modern computer science is based on the exponential amplification of computing power governed by the Moore's law. By increasing the number of transistors on a chip, while reducing its surface area, quantum effects are inevitable [1]. Unlike the deterministic classical systems (von Neumann computers), quantum technology is derived from quantum mechanics, a fundamental part of quantum field theory with its roots woven into the early quantum hypothesis made by Max Planck in 1900. The quantum hypothesis later blossoms into the wave-particle duality theory with limits to how accurately one can determine the position and the momentum of the observed particle. This uncertainty, in its most basic form, states that the product of the position uncertainty and the momentum uncertainty of the observed particle at the same time is greater than or equal to reduced Planck's constant, $\hbar = h/2\pi$:

$$\Delta p \cdot \Delta x \geqslant \hbar \tag{1}$$

The Heisenberg uncertainty principle, introduced in [2], represents theoretical extension to Copenhagen probabilistic philosophy, whose founding fathers are Bohr and Heisenberg, which states that physical systems generally do not have definite properties prior to being measured, and quantum mechanics can only predict the probability distribution of a given measurement's possible results [3]. Basically, the uncertainty principle addresses the relationship between what is (quantum states) and what can be measured (an observable quantity).

Emerging quantum problems in integrated circuits, microprocessors, quantum computing and quantum chemistry are most often analytically unsolvable, and in order to achieve accurate results, complex numerical models have to be employed. Instead of mesh-based numerical solvers, neural networks with as few as one hidden layer using nonlinear activation functions can serve as an accurate approximation to any continuous function, provided sufficiently many hidden units are available [4]. Lagaris et al. in [5] use feedforward neural networks with a single hidden layer as universal function approximators creating the framework for solving multiple problems in the area of quantum mechanics. Shirvany et al. in [6] show how a deep neural network is able to learn the wave function by minimizing the energy function. Only recently physics-informed neural networks (PINNs), introduced by Raissi et al. in [7], have reached significantly enhanced performance (the convergence speed and the accuracy of the final prediction) by exploiting features of the automatic differentiation [8].

This paper deals with a rather simple one-dimensional (1-D) eigenvalue problem, which serves as an opener to the subject. Here the combination of the ideas outlined in [9] and modern software capabilities of automatic differentiation [10] through Python and NumPy programming stack [11] are used to solve the particle-in-a-box problem. The paper is organized as follows: Section II represents the formulation of the problem and the standard analytical approach; in Section III, an overview of the neural network based numerical model is outlined; Section IV is dedicated to computational results and finally, conclusion and discussion, as well as the future work are presented in Section V.

II. Analytic approach to the particle-in-a-box quantum model

In classical physics one tackles with particles and fields, while in quantum physics wave function is considered. Thus, in classical physics there are particles interacting with different forces carried by fields (e.g. point charges and electromagnetic forces and fields in classical electromagnetism), while in quantum physics there is only wave function which, when observed, shows particle-like behaviour (the collapse of the wave function). Therefore, the task in classical physics is to prescribe a specific location and velocity of a particle and to

study how the particle evolves in accordance to the Newton mechanics. In quantum physics a particle is assigned the specific wave function and the evolution of such a system is governed by the Schrödinger equation.

When dealing with nonrelativistic velocities from classical mechanics, it follows that the total energy, E, of a particle of mass, m, is simply given as the sum of its kinetic and potential energy, U, respectively, i.e. it can be written as:

$$\frac{p^2}{2m} + U = E \tag{2}$$

where the value of the linear momentum of a particle is

$$p = mv \tag{3}$$

Replacing linear momentum with the corresponding quantum mechanical operator and introducing the wave function, ψ, one obtains the 3-D nonrelativistic Schrödinger equation:

$$-\frac{\hbar^2}{2m}\nabla^2\psi + U\psi = E\psi \tag{4}$$

As observable quantities are represented by mathematical operators, using the Hamiltonian operator, H, the Schrödinger equation (4) can be written as follows:

$$H\psi = E\psi \tag{5}$$

where the 3-D Hamiltonian operator is given by:

$$H = -\frac{\hbar^2}{2m}\nabla^2 + U \tag{6}$$

Note that the wave function, ψ, entirely describes the state of a quantum system. Since the particle must be found somewhere in space, Ω, it follows:

$$\int_\Omega \psi\psi^* d\Omega = \int_\Omega |\psi|^2 d\Omega = 1 \tag{7}$$

Functions satisfying (5) are referred to as eigenfunctions of operator H, while corresponding values of energies are eigenvalues. Thus, if a certain wave function is an eigenfunction of H, then it follows:

$$H\psi_n = E_n\psi_n \tag{8}$$

with the related eigenvalue, E_n. Furthermore, as the mean value of an observable is the expectation value of the related operator, according to the notation from statistical analysis point of view, ψ_n, the expectation of an operator A is given by:

$$\langle A_n \rangle = \frac{\int_\Omega \psi^* A\psi d\Omega}{\int_\Omega \psi^*\psi d\Omega} \tag{9}$$

A simplified case of the 1-D time-independent Schrödinger equation for a single nonrelativistic particle can then be written in the following manner:

$$H\psi(x) = E\psi(x) \tag{10}$$

where the corresponding Hamiltonian operator has the following form:

$$H = -\frac{\hbar^2}{2m}\frac{d^2}{dx^2} + U(x) \tag{11}$$

The wave function squared, $|\psi(x)|^2$, is the probability density function (PDF) of the particle along the x-axis. Due to the Heisenberg uncertainty principle the actual position of a particle is not known, rather one can describe the probability of finding the particle for position in $[x, x + \mathrm{d}x]$. Therefore, the wave function can be viewed as a mathematical tool to calculate the probability of observing the particle in a specific position, or simply the quantity which squared value yields the probability of observing the corresponding outcome. Physically observable system is presented in a measurable and meaningful fashion, provided the wave function satisfies the following set of constraints:

- The wave function must be a solution to the Schrödinger equation;
- The wave function and the first derivative of the wave function, $d\psi(x)/dx$, must be continuous;
- The wave function must be normalizable – the wave function value approaches to zero as x approaches infinity.

Considering the particle-in-a-box problem, an electron of mass m_e is confined to a 1-D rigid, infinite potential well of width L, and following the previously outlined constraints it can be shown that the wave function at the ground state is defined using the following expression:

$$\psi(x) = \sqrt{\frac{2}{L}}\sin\left(\frac{\pi}{L}x\right) \tag{12}$$

The potential of the infinite potential well is defined as follows:

$$U(x) = 0, \qquad x \in \langle 0, L\rangle \tag{13}$$

$$U(x) = +\infty, \qquad x \in \langle -\infty, 0] \cup [L, +\infty\rangle \tag{14}$$

An electron is able to move along x-axis and the collisions with the walls are considered perfectly elastic. The full mathematical description can be found elsewhere, e.g. in [3].

In Fig. 1, the analytic solution of the wave function and the PDF of an electron in the infinite potential well are shown.

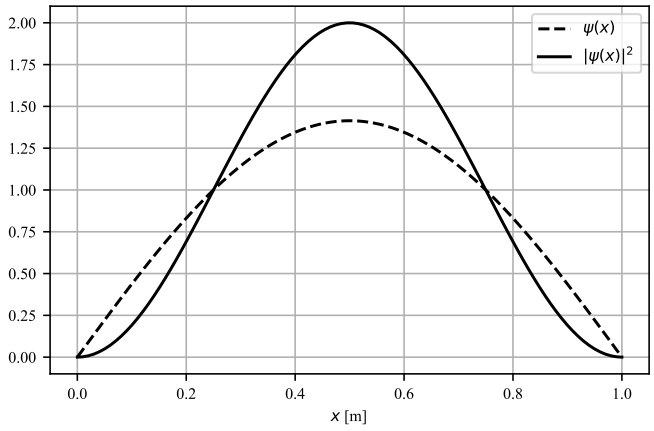


Fig. 1. The analytic solution of the wave function and the probability density function for the case of the 1-D particle-in-a-box problem.

The highest point of the PDF represents the most probable position of an electron at any given moment in time. The term

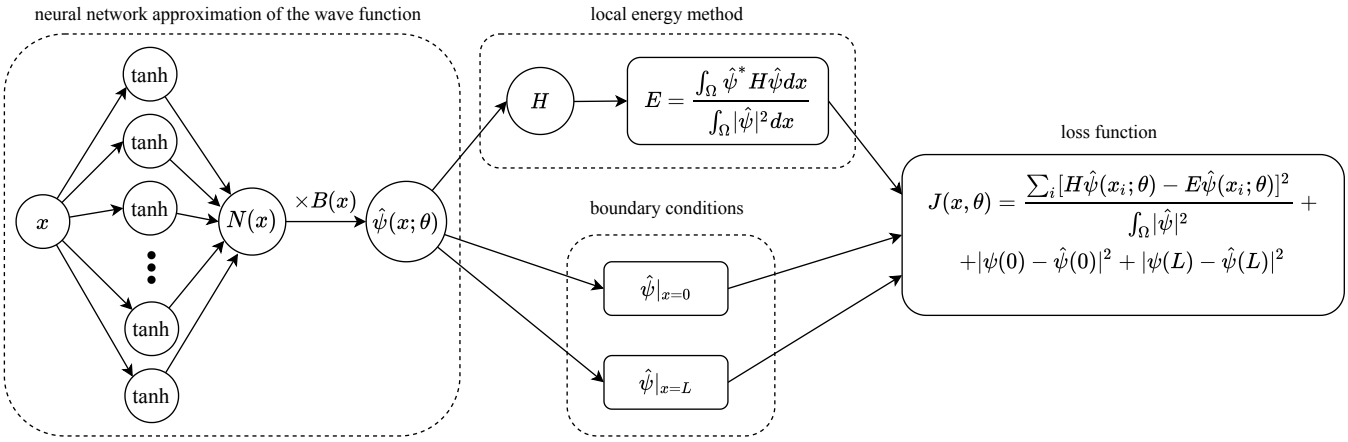


Fig. 2. An artificial neural network configuration for solving eigenvalue problems with known boundary conditions. The output of a neural network is scaled using the design function $B(x)$ and used to construct the loss function to be minimized and to obtain optimal parameters, θ, of a neural network.

$\sqrt{2/L}$ in (12) represents the amplitude of the wave function and if it decreases, the probability density peak will decrease, which will subsequently cause the PDF curve to be flatten.

III. Artificial Neural Network Numerical Solution

The technique proposed in [9] is extended and adjusted to modern machine learning practices in order to acquire the wave function in the particle-in-a-box quantum model. Firstly, the wave function is approximated using the following expression:

$$\hat{\psi}(x) = B(x)N(x;\theta) \tag{15}$$

where $B(x)$ is a design function that satisfies boundary conditions, in this case $B(x) = x(1-x)$, and $N(x;\theta)$ is the neural network that takes positional coordinate x as the input argument and is parameterized using the weights and biases, θ. The key idea is to solve the Schrödinger equation by representing the solution using the neural network and training the resulting network to satisfy the conditions required by the equation itself. The collocation method is applied to discretize the calculation domain, Ω, into the set of points for training, $x_i \in \Omega$ for $i \in [1, N]$, where N is the total number of collocation points. The problem is transformed into the optimization problem with the following loss function:

$$J(x,\theta) = \frac{\sum_i [H\hat{\psi}(x_i;\theta) - E\hat{\psi}(x_i;\theta)]^2}{\int_\Omega |\hat{\psi}(x_i;\theta)|^2 dx} + |\psi(0) - \hat{\psi}(0;\theta)|^2 + |\psi(L) - \hat{\psi}(L;\theta)|^2 \tag{16}$$

where

$$E = \frac{\int_\Omega \hat{\psi}^* H\hat{\psi} dx}{\int_\Omega |\hat{\psi}|^2 dx} \tag{17}$$

The formation of the loss function (16) is visualized in Fig. 2. Just like in the original work, the neural network is the single hidden layer multi-layer perceptron (MLP) with 40 hidden units total. Every unit is activated using hyperbolic tangent, *tanh*, activation function. Instead of manually applying symbolic derivatives, the automatic differentiation procedure is applied. Once the derivative of the loss function with respect to the network parameters has been acquired, the minimization can be performed. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm, an iterative method for solving unconstrained nonlinear optimization problems [12], is applied.

IV. Results

Using the network architecture defined in Section III, three test cases are executed, where the neural network architecture has remained constant, only the number of collocation (training) data points have differed. The values of the root mean squared error (RMSE) for the neural network are given in Table I, together with the RMSE of finite element method (FEM) and finite difference method (FDM). Mathematical

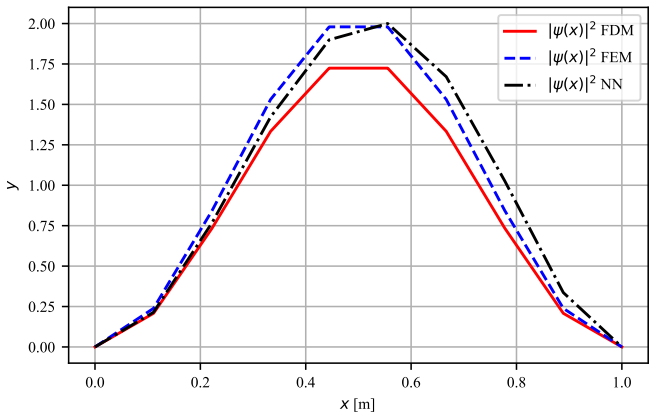
TABLE I
Comparison of the Neural Network Accuracy with Finite Difference and Finite Element Numerical Methods

Collocation points	Root Mean Square Error		
	Finite Difference	Finite Element	Neural Method
10	0.12910	0.02384	0.08633
50	0.02474	0.00083	0.05863
100	0.01231	0.00020	0.03783

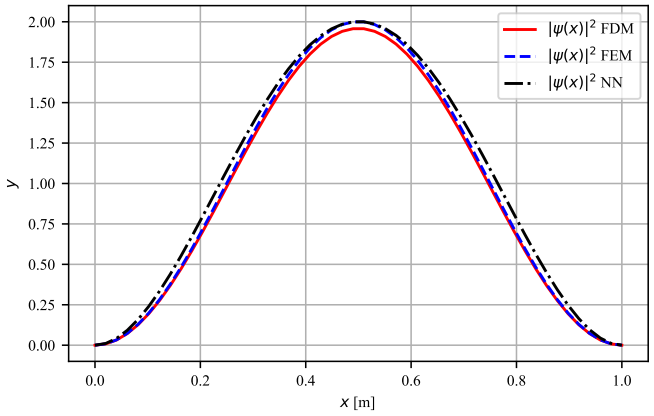
details of FEM and FDM implementation can be found elsewhere, e.g. in [13]. The constructed neural network achieves the best results when the tolerance for termination of the BFGS optimization algorithm is set sufficiently low. The number of training epochs has been set to 2000 but due to the nature of the loss function, all three test cases converged to accurate solution in under 100 iterations. FEM achieves the best results for all three test cases and the neural network performs more satisfactorily than FDM when the number of collocation points is 10. For 50 and 100 collocation points the accuracy of FDM surpasses the accuracy of the neural network for more than 3 times. This can be attributed to the increase of the space

of parameters that need to be optimized by BFGS and is possible to improve using advanced optimization techniques, more suitable for neural networks.

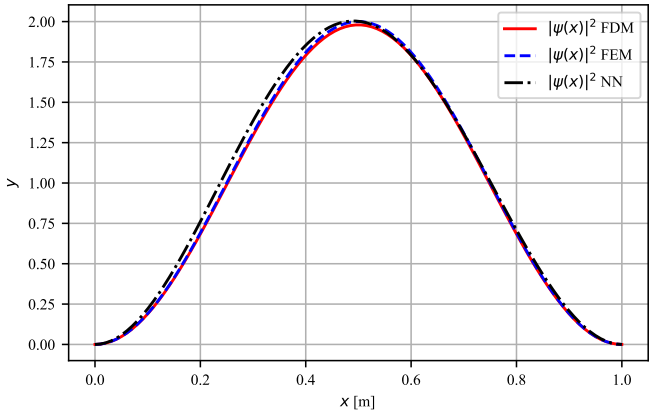
The family of wave functions obtained via FDM, FEM and the neural network is shown in Fig. 3a, Fig. 3b and Fig. 3c for 10, 50 and 100 collocation points, respectively.



(a)



(b)



(c)

Fig. 3. Wave functions approximated with 10 in Fig. 3a, 50 in Fig. 3b and 100 collocation points in Fig. 3c. In each figure, FDM, FEM and the neural networks are shown using red, blue and black line, respectively.

The calculation of RMSE is repeated for the range of collocation points between 10 and 100 for all three numerical methods. The graphical presentation of the relationship between the RMSE and the number of collocation points is shown in Fig. 4. Increasing the number of collocation points

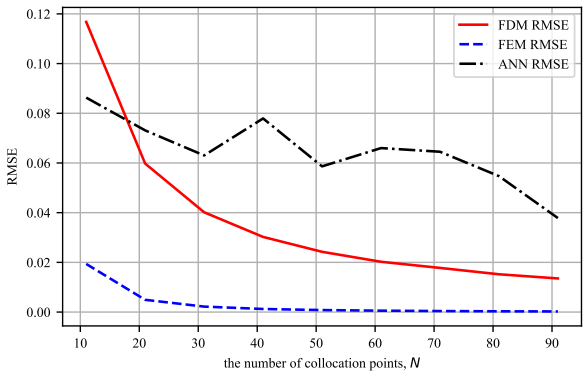


Fig. 4. RMSE values for FDM, FEM and the neural network for the range of collocation points between 10 and 100. The random nature of the neural network RMSE is a direct consequence of the very core of the method itself that lies in the random initialization of neural network weights based on the Xavier scheme [14].

in the solution domain, both FEM and FDM give increasingly accurate solutions with a relatively slight slowdown in calculation due to the simplicity of the problem. The neural network approach, however, does not provide a significant increase in accuracy by increasing the number of input data space and although both mesh-based methods surpassed the accuracy of the neural network solver, it should be noted that the accuracy of the neural network is comparable to the FDM for the entire range of collocation points. It is also worth noting that a single important advantage of the neural network approach is the interpolation capabilities inherently embedded in the method itself. Both FEM and FDM, and any other mesh-based numerical method, require additional interpolation computations in order to find the value of the solution at an arbitrary point in the domain. On the other hand, both FEM and FDM are rigorously derived from the governing Schrödinger equation, which makes them more reliable in practice and in theory, respectively. The neural network approach in this case, although minimizing the Schrödinger equation residual is still a *black box* model.

V. Conclusion

In this paper the interpolation capabilities of the neural network eigenvalue problem solver are demonstrated on the simple example of particle-in-a-box. Using the compact shallow neural network, one is able to achieve highly accurate results comparable to both finite element and finite difference methods, respectively. Possible progress can be achieved using more complex neural network architectures and more advanced optimization techniques. However, this simple case shows the potential advantages of the neural network approach,

in its nature adaptable to parallel hardware and resistant to high dimensionality.

ACKNOWLEDGMENT

This research has been funded by DATACROSS project of The Centre of Research Excellence for Data Science and Advanced Cooperative Systems (CRE ACROSS-DataScience).

REFERENCES

[1] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1994, pp. 124–134.

[2] W. Heisenberg, "Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik," *Zeitschrift für Physik*, vol. 43, pp. 172 – 198, 1927. [Online]. Available: https://doi.org/10.1007/BF01397280

[3] D. Griffiths and D. Schroeter, *Introduction to quantum mechanics*. Cambridge University Press, 2018.

[4] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359 – 366, 1989. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0893608089900208

[5] I. Lagaris, A. Likas, and D. Fotiadis, "Artificial neural network methods in quantum mechanics," *Computer Physics Communications*, vol. 104, no. 1, pp. 1 – 14, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0010465597000544

[6] Y. Shirvany, M. Hayati, and R. Moradian, "Numerical solution of the nonlinear Schrödinger equation by feedforward neural networks," *Communications in Nonlinear Science and Numerical Simulation*, vol. 13, no. 10, pp. 2132 – 2145, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1007570407001037

[7] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686 – 707, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999118307125

[8] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: A survey," *Journal of Machine Learning Research*, vol. 18, p. 5595 – 5637, 2017. [Online]. Available: https://dl.acm.org/doi/10.5555/3122009.3242010

[9] I. Lagaris, A. Likas, and D. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 987 – 1000, 1998. [Online]. Available: https://ieeexplore.ieee.org/document/712178

[10] D. Maclaurin, D. Duvenaud, and R. P. Adams, "Autograd: Effortless gradients in Numpy," *ICML 2015 AutoML Workshop*, 2015.

[11] T. Oliphant, "Python for scientific computing," *Computing in Science and Engineering*, vol. 9, pp. 10–20, 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4160250

[12] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer New York, 2006. [Online]. Available: https://doi.org/10.1007/978-0-387-40065-5_6

[13] D. Poljak, S. Šesnić, A. Rubić, and E. Maze, "A note on the use of analytical and domain discretisation methods for the analysis of some phenomena in engineering physics," *International Journal for Engineering Modelling*, vol. 31, no. 1 - 2, pp. 43 – 60, 2018. [Online]. Available: https://hrcak.srce.hr/206316

[14] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 9, 2010.